PATENT ABSTRACTS OF JAPAN

(11)Publication number : 03-164835
(43)Date of publication of application : 16.07.1991

(51)Int.Cl. G06F 9/45

(21)Application number : 01-302149
(22)Date of filing : 22.11.1989

(71)Applicant : HITACHI LTD
HITACHI TOHOKU SOFTWARE KK

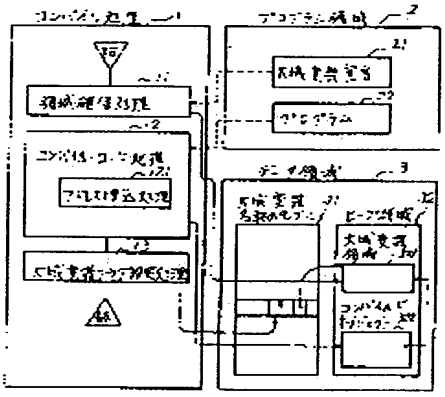
(72)Inventor : HIRAIDE TOSHIMICHI
HIROSE TADASHI
HARA MIYUKI
TOMONAGA KATSUKO

(54) COMPILING METHOD FOR GLOBAL VARIABLE PROCESSING IN INTERPRETER TYPE LANGUAGE PROCESSING SYSTEM

(57)Abstract:

PURPOSE: To increase the execution speed of a compiled object by performing the processing, which interprets a specification statement to reserve a variable area of a global variable in accordance with the type and the size of the variable, and the processing, which sets the address of the reserved variable area to program areas where the global variable is referred and updated, at the time of area reserving processing.

CONSTITUTION: An area reserving processing 11, an address setting processing 121 in a compile load processing 12, and a global variable flag setting processing 13 are provided. The area reserving processing 11 reserves a global variable area in accordance with a global variable declaration 21 included in a program area 2 and stores its address. A compiled object 322 where the address of a processing object (the address in a global variable area 321) is reserved for the global variable in a program 22 is generated based on the stored address. Thereafter, a change inhibition flag of the global variable name as the object in global variable name table 31 is set to inhibition by the global variable flag setting processing 13. Thus, the execution speed of the compiled object of the program is increased.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

⑲日本国特許庁（JP）　⑪特許出願公開

⑫公開特許公報（A）　平3－164835

⑤1Int.Cl.⁵　識別記号　庁内整理番号　⑬公開　平成3年(1991)7月16日

G 06 F　9/45

8724－5B　G 06 F　9/44　322　F

審査請求　未請求　請求項の数 3　（全7頁）

⑤4発明の名称　インタプリタ型言語処理系における大域変数処理のコンパイル方法

⑳特　願　平1－302149

㉒出　願　平1(1989)11月22日

⑦2発明者　平出　利道　宮城県仙台市青葉区一番町2丁目4番1号　日立東北ソフトウエア株式会社内

⑦2発明者　広瀬　正　神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

⑦2発明者　原　幸　神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア工場内

⑦1出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地

⑦1出願人　日立東北ソフトウエア株式会社　宮城県仙台市青葉区一番町2丁目4番1号

⑦4代理人　弁理士　小川　勝男　外1名

最終頁に続く

明　細　書

1．発明の名称

インタプリタ型言語処理系における大域変数処理のコンパイル方法

2．特許請求の範囲

1．大域変数処理機能を持つインタプリタ型言語処理系において、予めコンパイル対象となるソースプログラムと共に定義された大域変数の名称と変数の型と変数の大きさに従って大域変数の領域を静的に確保する処理（領域確保処理）をし、大域変数を参照・更新するプログラム領域に領域確保処理で確保した領域のアドレスを埋め込む処理（アドレス埋込処理）を行ないながらコンパイル対象プログラムも機械語形式のプログラム（コンパイルドオブジェクト）に変換し、データ領域に書き込み処理（コンパイル・ロード処理）を行なうことを特徴としたインタプリタ型言語処理系における大域変数処理のコンパイル方法。

2．請求項1記載の方法において、大域変数名称のテーブルに変更不可フラグ領域を設定し、コンパイル・ロード処理後にアドレス埋込処理を行なつた大域変数名称の変更不可フラグを不可に設定し、コンパイル対象外のプログラムから呼び出された大域変数の変数領域の生成・削除処理あるいは変数領域の参照・更新処理の発生時に変更不可フラグを判定し、不可ならばその生成・削除処理あるいは参照・更新処理を異常終了し、不可以外ならばその処理を行うことを特徴とするインタプリタ型言語処理系における大域変数処理のコンパイル方法。

3．請求項1記載の方法において、大域変数領域の領域確保処理時に大域変数名称のテーブルの変更不可フラグを判定し、不可でなくかつ確保する領域と同一の大きさを持つ領域が既存するならば領域確保処理を行なわずに既存の領域を再利用することを特徴とするインタプリタ型言語処理系における大域変数処理のコンパイル方法。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明は、インタプリタ型言語処理系の処理方法に関する。

〔従来の技術〕

例えばインタプリタ型言語処理系の１つであるPROLOG処理系には、例えば、「日立クリエーティブワークステーション２０５０，プロローグ（株式会社日立製作所ソフトウエア工場，昭和６３年６月発行）の第８７頁」に論じられているように大域変数（グローバル変数）が用意されている。しかしながら、この種の大域変数はそのものが動的に生成・削除されること、デバッグ終了部分から順次プログラムを部分的にコンパイルすることができるインタプリタ型言語開発環境の特徴を確保するため、従来はコンパイル対象プログラムで大域変数を参照・更新を機械語形式のプログラムに変換していなかつた。従つて大域変数の参照・更新処理は、実行時に大域変数の変数領域のアドレスを大域変数名称のテーブルから取り出していた。

〔発明が解決しようとする課題〕

上記従来技術は、コンパイル対象プログラム内に大域変数を参照・更新するプログラム部分があるとそこはインタプリテイブに実行せざるをえず、コンパイルドオブジエクト全体の実行速度に多大な影響を与えるもので、コンパイルドオブジエクトの実行速度の高速化を妨げているという問題があつた。

本発明の目的は、大域変数処理を含むプログラムのコンパイルドオブジエクトの実行速度の高速化を図ることにある。

〔課題を解決するための手段〕

上記目的を達成するために、本発明においては大域変数の名称と変数の型と変数の大きさを記述する宣言文を用意し、領域確保処理時にその宣言文を解釈して変数の型と変数の大きさから大域変数の変数領域を確保する処理と、大域変数を参照・更新するプログラム領域に上述で確保した変数領域のアドレスを埋め込む処理をすることで、従来コンパイルドオブジエクトの実行時に行つていた大域変数名称のテーブルから変数領域のアドレスを取り出す処理を不要とし、コンパイルドオブジエクトの実行速度の高速化を実現する。

また、大域変数名称のテーブルに変更不可フラグ領域を設定し、アドレス埋込処理を行なつた大域変数名の変更不可フラグを不可に設定し、コンパイル対象外のプログラムからの大域変数の変数領域の生成・削除処理あるいは参照・更新処理に対して変更不可フラグを参照し、不可ならばその処理を異常終了し、さもなければその処理を実行する。

さらに、領域確保処理時に宣言文による大域変数の名称から変更不可フラグを参照し、不可以外でかつ確保する領域と同一の大きさを持つ領域が存在するならば既存の領域を再利用する。

〔作用〕

大域変数の領域確保処理時には、大域変数の宣言文中の変数の大きさから確保する変数領域の大きさと確保した変数領域のアドレスを求めることができ、コンパイル対象プログラムで大域変数領域を参照・更新する部分は、参照・更新するアドレスが決定できた機械語形式のプログラムに変換することができる。それによつて、従来コンパイルドオブジエクトの実行時に大域変数名称のテーブルから変数領域のアドレスを取り出し、参照・更新するアドレスを決定していた処理が不要となり、コンパイルドオブジエクトの実行速度の高速化が実現できる。

また、大域変数名称のテーブルに変更不可フラグ領域を設定し、コンパイル対象となつた大域変数の変更不可フラグ領域のフラグを不可に設定しておき、さらにコンパイル対象外のプログラムからの大域変数領域の生成・削除処理あるいは参照・更新処理に対して変更不可フラグ領域のフラグを判定することにより、フラグが不可ならばその処理が異常終了でき、さもなければその処理が実行できる。したがつて、コンパイル対象プログラムを実行プログラムの一部に限定した場合でも、正しくない動作の発生を防止できる。なぜなら、コンパイル対象プログラム内の大域変数に対して

確保した変数領域を、コンパイル対象外のプログラムからの生成・削除処理あるいは参照・更新処理で、破壊される場合は、上記異常終了を発生させ、その旨をプログラム作成者に知らせることができるからである。

さらに、前述の宣言文による大域変数の名称と変数領域の大きさに基づき、コンパイル対象外のプログラムの当該大域変数にたいする生成・削除処理時で、大域変数名称のテーブルの変更不可フラグを処理実行前に判定し、もし変更不可フラグが設定されているとき、領域確保処理時には新規に変数領域を確保することはせず、また領域削除処理時には領域を削除しないことにより、コンパイル対象プログラムを実行プログラムの一部分に限定しても従来コンパイルした部分と同等の動作を保証することができる。

〔実施例〕

第１図は、本発明による大域変数処理を含むプログラムのコンパイル処理の概要を表すフローチャートである。図において点線は情報の流れを表し、実線は処理の流れを表す。

コンパイル処理とは、プログラム領域２内に格納されるプログラム２２をコンパイル・ロード処理１２で解析・変換処理を行つた結果をデータ領域３内のヒープ領域３２にコンパイルドオブジエクト３２２として生成するものである。従つて、従来のデータ領域３には大域変数名称のテーブル３１とヒープ領域に含まれるコンパイルドオブジエクト３２２が存在する。

本発明では、従来のコンパイル処理に領域確保処理１１とコンパイル・ロード処理１２内のアドレス埋込処理１２１と大域変数フラグ設定処理１３を設けたことに特徴がある。第１図を用いて各処理の概要を説明する。

領域確保処理１１はプログラム領域２内に含まれる大域変数宣言２１から大域変数領域を確保し、そのアドレスを格納する。プログラム２２中にある大域変数は、前述で格納したアドレスから処理対象のアドレス（大域変数領域３２１内のアドレス）を確定したコンパイルドオブジエクト３２２を生成する。

その後、大域変数フラグ設定処理１３で大域変数名称のテーブル３１の対象となつた大域変数名称の変更不可フラグを不可に設定する。

第２図は、第１図の大域変数名称のテーブルの詳細と大域変数名称のテーブル３１と大域変数領域３２１の関係を示した図である。この図を用いて領域確保処理１１と大域変数フラグ設定処理１３を詳述する。

大域変数名称のテーブル３１の１エントリーは以下の５つのフイールド、すなわち大域変数名称３１１と変更不可フラグ領域３１３と大域変数領域３２１のアドレス領域３１４と他の領域３１２，３１５とからなる。３１２，３１５は、本発明と直接関係ないので説明を省略する。

領域確保処理１１では、大域変数の宣言（大域変数名称，変数の型，変数の大きさ）の変数の型と変数の大きさに基づきプログラム領域２の中に変数領域を確保し、そのアドレスを該当する大域変数に対応する１つのエントリの大域変数名称のアドレス領域３１４に格納する。

大域変数フラグ設定処理１３は、変数領域321を確保した大域変数名称の変更不可フラグ領域３１２を不可とする処理である。

第３図は、第１図のコンパイルドオブジエクト３２２と大域変数領域３２１との関係を示した図である。この図を用いてアドレス埋込処理１２１を詳述する。

大域変数処理２２１を含むプログラム２２のコンパイル・ロード処理１２において、大域変数領域３２１のアドレスは既に領域確保処理１１で変数の確保をしたときに決定されており、大域変数名称のテーブルのアドレス領域３１４に格納されている。そこで、アドレス埋込処理１２１は、大域変数処理２２１の対象アドレスを求めコンパイルドオブジエクト３２２（機械語形式のプログラム）に組み入れる。

第４，５図は、前述のアドレス埋込処理１２１で生成される機械語形式のプログラムへの変換例を示したものである。

第４図は、大域変数領域の更新処理プログラム４１１を含むプログラム４１が、アドレス埋込処理１２１により機械語形式のプログラム４２内の命令４２１に変換される例を示したものである。

global_nameは大域変数の名称を示し、Indexは変数領域内の対象となるフイールドを指定するインデツクスを示し、＿は変数領域内の対象フイールドの値（非更新値）を判定しないことを示し、１は更新値を示す。すなわち、更新処理プログラム例４１のglobal_update(global_name, Index,＿,1)４１１は、大域変数名称global_nameのIndex番目のフイールドの値を整数値の１に変更する命令である。この処理が機械語形式のプログラム４２では、MOVE #1,global_name_addr+Index４２１に変換できる。ここでglobal_name_addrは、領域確保処理１１で確保した変数領域のアドレスを示し、global_name_addr+Indexは変数領域内の対象フイールドのアドレスを示す。すなわちこの変換処理において、アドレス埋込処理１２１は、領域確保処理１１で確保した変数領域のアドレス(global_name_addr)を大域変数名称のテーブル３１のアドレス領域３１４から求めることが出来るため、変数領域内の処理対象フイールドのアドレス(global_name_addr+Index) を求め、機械語形式のプログラムの命令４２１に変換する。

第５図は、大域変数領域の参照処理プログラム５１１を含むプログラム５１が、アドレス埋込処理１２１により機械語形式のプログラム５２内の命令５２１に変換される例を示したものである。

global_nameは大域変数の名称を示し、Indexは変数領域内の対象となるフイールドを指定するインデツクスを示し、１は参照値を示す。すなわち、参照処理プログラム例５１のglobal_unify(global_name, Index,1) ５１１は、大域変数名数global_nameのIndex番目のフイールドの値が整数値の１かどうか判定（参照）する命令である。この処理が機械語形式のプログラム５２では、COMP #1, global_name_addr+Index５２１,BNE FAIL ５２２に変換できる。ここでglobal_name_addrは、領域確保処理１１で確保した変数領域のアドレスを示し、global_name_addr+Indexは変数領域内の対象フイールドのアドレスを示し、ＦＡＩＬは判定（参照）が失敗したときに実行するアドレスを示すが、本発明と直接関係ないため詳細の説明は省略する。すなわちこの変換処理において、アドレス埋込処理１２１は、領域確保処理１１で確保した変数領域のアドレス（global_name_addr）を大域変数名称のテーブル３１のアドレス領域３１４から求めることが出来るため、変数領域内の処理対象フイールドのアドレス（global_name_addr+Index）を求め、機械語形式のプログラムの命令５２１～５２２に変換する。

第６図は、変更不可フラグを用いた大域変数の生成・削除処理フローを示している。

変数の大きさが正の時に、大域変数の生成処理６１を行う。大域変数名称のテーブルの対応する変数名称の１フイールド内の変更不可フラグ領域３２１のフラグを判定しフラグが不可、つまり変数領域が使用中の場合は生成処理を異常終了611する。さもなければ、変数領域が未使用とみなし変更不可フラグ領域３２１のフラグを不可にする処理６１２を行う。

また、変数の大きさが正以外の場合は大域変数の削除処理６２を行う。削除処理６２は、大域変数名称のテーブルの対応する変数名称の１フイールド内の変更不可フラグ領域３２１のフラグを判定し、フラグが不可、つまり変数領域が使用中の場合は、確保した変数領域はそのままにし変更不可フラグ領域３２１のフラグを不可以外にする６２１処理を行う。さもなければ変数領域が未使用とみなし、削除処理６２を異常終了６２２する。

第７図は、コンパイル対象プログラム内の大域変数と非コンパイル対象プログラム内の大域変数の処理を示した図である。

コンパイル対象プログラム７１は、大域変数の宣言７１１，７１２と大域変数生成処理７１３，７１４と大域変数の更新処理７１５，７１６を含むプログラムである。非コンパイル対象プログラム７２は、大域変数生成処理７２１，大域変数更新処理７２２，大域変数参照処理７２３を含むプ

ログラムである。

ここで、コンパイル対象プログラムをコンパイル処理するとデータ領域７３には、大域変数名称のテーブル７３１内のフィールド７３１１に大域変数名ａｂｃを、同内のフィールド７３１２に大域変数名ｃｄｅを登録し、ａｂｃの変数領域732とｃｄｅの変数領域７３３を確保し、ａｂｃの変更不可フラグ領域７３１３のフラグとｃｄｅの変更不可フラグ領域７３１４のフラグを不可に設定する。

この図は、非コンパイル対象プログラム７２をインタプリテイブに実行させると、大域変数の生成，更新，参照の各処理は、変数名称ｃｄｅの変更不可フラグ領域７３１４のフラグを判定し、不可であることからコンパイル対象プログラム７１により変数領域７３３が使用中であると判断し、異常終了することを示している。

第８図は、変更不可フラグを用いた領域確保処理１１の詳細フローである。

宣言文（大域変数名称，変数の型，変数の大きさ）の解析処理８１を行ない変数の型と変数の大きさに基づき、確保する変数領域の大きさを求める処理８２をし、領域の確保を行う。このとき、大域変数名称のテーブルのフィールドの変更不可フラグ領域のフラグが不可以外でかつ確保する変数領域と大きさが等しい領域が存在する場合に、既存の変数領域が再利用可能となり、再利用処理８３を行なう。再利用処理８３では、大域変数名称のテーブル内の対象フィールドの変数名称を変更する処理８３１を行なう。さもなければ、新たに変数領域を確保する処理（新規変数領域の確保処理８４）を行なう。新規変数領域の確保処理８４では、変数領域の確保処理８４１をし、大域変数名称のテーブル内に新たなフィールドを設定し、そのフィールド内に大域変数名称を登録する処理（大域変数名称の登録処理８４２）をし、処理８４１で得たアドレスを同フィールド内のアドレス格納領域に格納する処理（変数領域アドレス格納処理８４３）を行なう。

〔発明の効果〕

本発明によれば、インタプリタ型言語処理系における大域変数処理方法において、コンパイルドオブジエクト生成時に大域変数の処理対象アドレスの決定が可能となる。

また、デバツク終了部分から順次、プログラムを部分的にコンパイルすることができるインタプリタ型言語のプログラム開発環境の特徴を従来のまま確保できる。

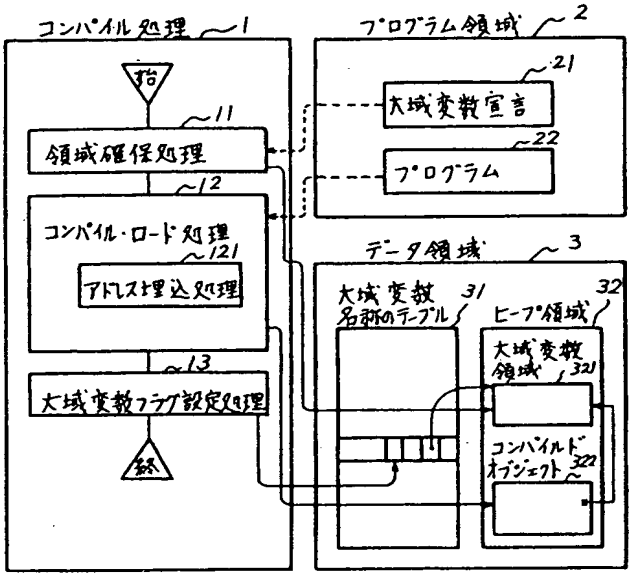
この結果、従来、コンパイルドオブジエクトの実行時に大域変数の処理対象アドレスを求めていたことに比較してより高速実行可能となる。

4．図面の簡単な説明

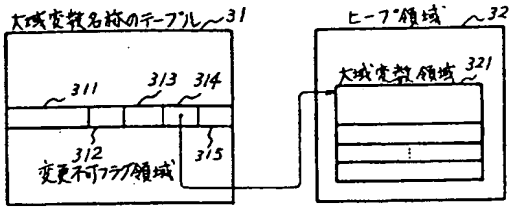
第１図は本発明の概要を示すコンパイル処理フロー図、第２，３図は本発明の一実施例の処理の詳細説明図、第４図は本発明の実施例の大域変数更新処理の機械語形式プログラムへの変換を示す説明図、第５図は本発明の実施例の大域変数参照処理の機械語形式プログラムへの変換を示す説明図、第６図は本発明の実施例の大域変数生成，削除処理フロー図、第７図は本発明の実施例のコンパイル対象プログラムと非コンパイル対象プログラム内の大域変数処理の関連の説明図、第８図は本発明の実施例の大域変数領域の確保処理フロー図である。

代理人　弁理士　小川勝男

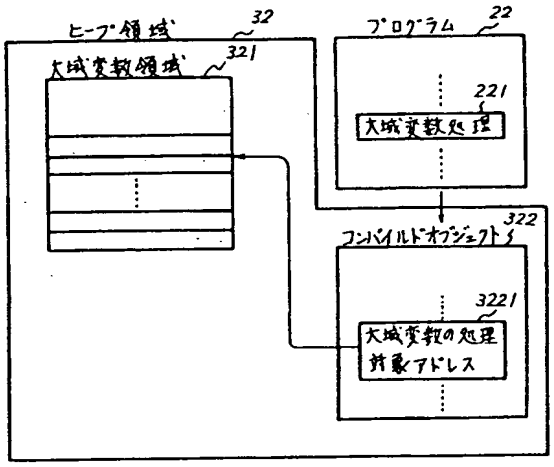
第 1 図



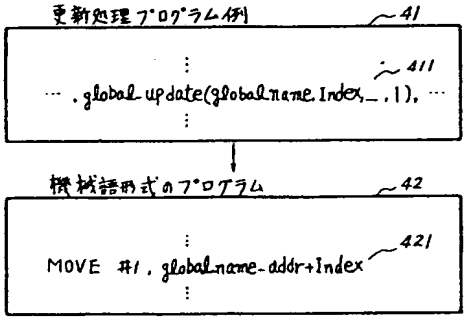
第 2 図



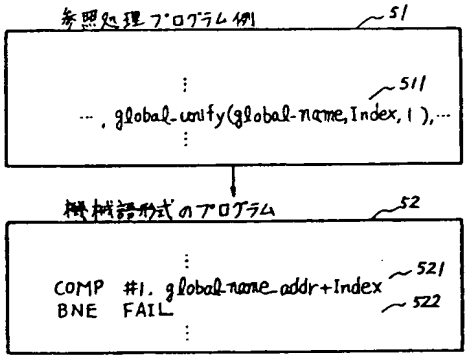
第 3 図



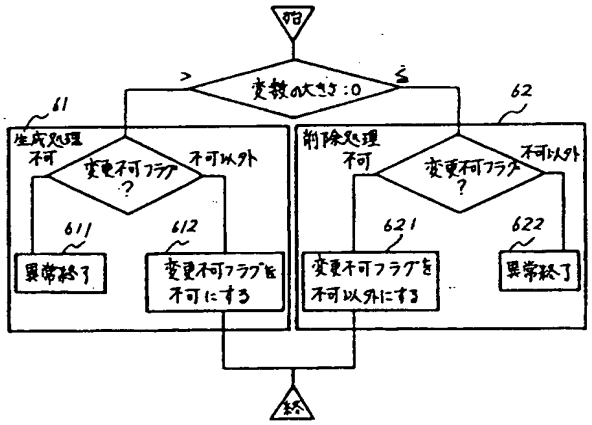
第 4 図



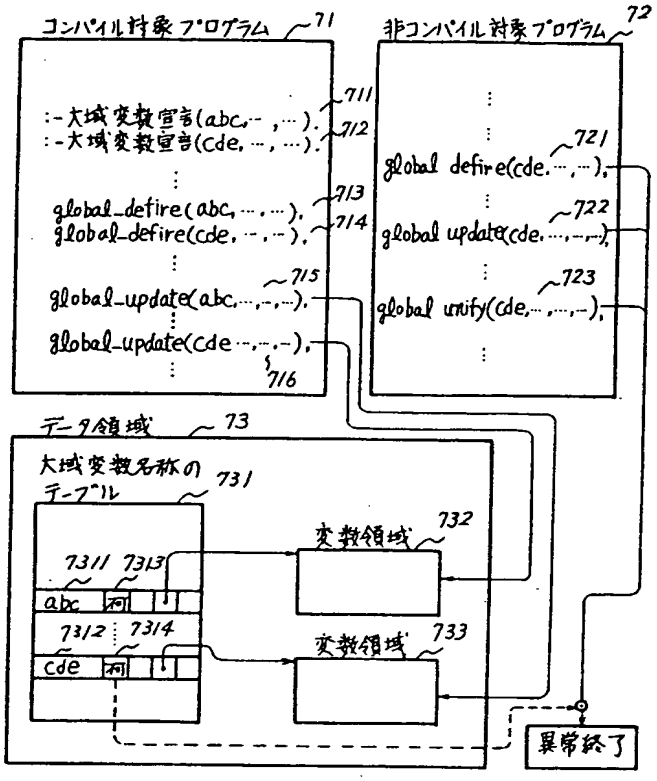
第 5 図



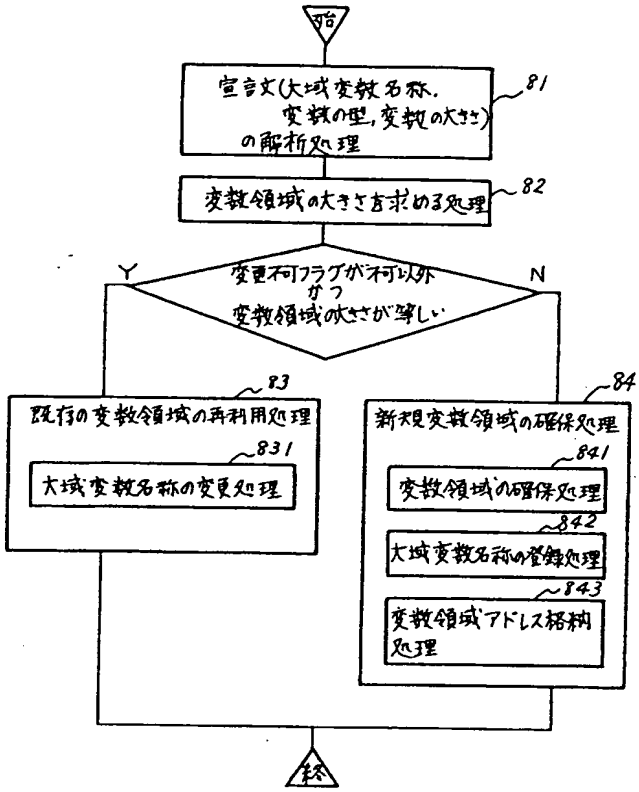
第 6 図



第 7 図



第 8 図



第1頁の続き

⑫発 明 者　友永　佳津子　神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア工場内